

Харківський університет радіоелектроніки

Факультет комп'ютерних наук

Кафедра програмної інженерії

ЗВІТ

до практичного заняття з дисципліни

"Аналіз та рефакторинг коду"

на тему: "Мова програмування C#. Code convention"

Виконав ст. гр ПЗП-22-2

Терновий Даніїл Павлович

Перевірів

доцент кафедри ПІ

Лещинський Володимир Олександрович

Харків 2024

## **МЕТА РОБОТИ**

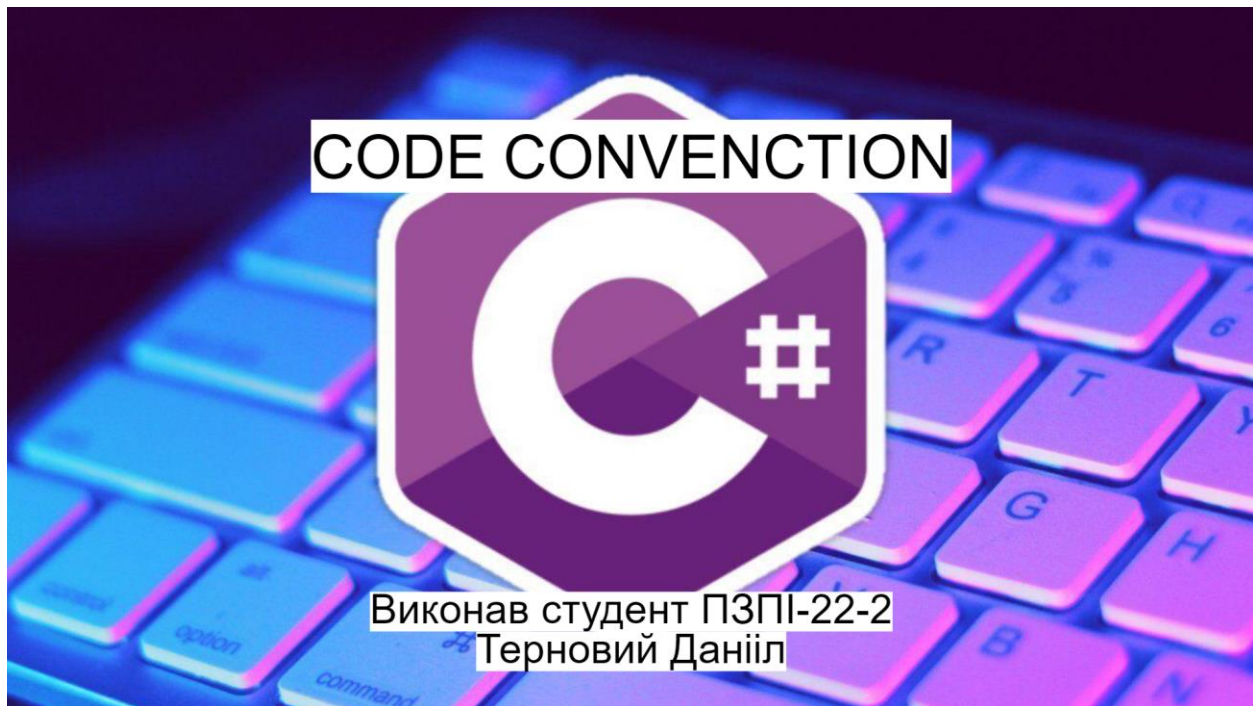
Ознайомитись із основними принципами написання чистого та структурованого коду на мові програмування C#, зосередившись на стандарті оформлення коду. Дослідити найкращі практики та рекомендації для C#, супроводжуючи їх прикладами коду.

## **ВИСНОВКИ**

У ході виконання завдання я ознайомився із ключовими принципами написання чистого та структурованого коду на мові програмування C#.

## ДОДАТОК А

Слайди презентації



### Стилі найменування

- Pascal case

Кожне слово в імені починається з великої літери без розділових знаків між ними

- Camel case

Перше слово починається з малої літери, а всі наступні слова — з великої

## Приклади стилів найменування

Pascal case

```
0 references  
public async Task<IActionResult> ConfirmEmail(string userId, string token)  
{  
}
```

Camel case

```
public AccountManagerController(  
    UserManager<CarCoolUser> userManager,  
    SignInManager<CarCoolUser> signInManager,  
    RoleManager<IdentityRole> roleManager,  
    ITokenService tokenService,  
    Services.IEmailSender emailSender,  
    IMediaObjectRepository mediaObjectRepository,  
    IConfiguration configuration,  
    ILogger<AccountManagerController> logger)  
{  
}
```

3

## Поширені практики наймінгу

Зазвичай я використовую наступні правила:

- Публічні методи та змінні в Pascal case
- Приватні властивості в Camel case з префіксом “\_”
- Інтерфейси в CamelCase з префіксом “I”

4



## Приклади

```
2 references
public class LotUpdateModel
{
    2 references
    public string LotId { get; set; }
    1 reference
    public int BidCount { get; set; }
    1 reference
    public decimal CurrentAmount { get; set; }
}
```

Публічні поля

```
private readonly AzureOptions _azureOptions;
private readonly IMediaObjectRepository _mediaObjectRepository;
private readonly IMapper _mapper;
```

Приватні поля

```
Interfaces
└─ IC# IBidRepository.cs
└─ IC# IBrandRepository.cs
└─ IC# ICommentRepository.cs
└─ IC# ICompletionRepository.cs
└─ IC# IDisadvantageRepository.cs
└─ IC# IEmailTemplateRepository.cs
└─ IC# IFavoriteLotRepository.cs
└─ IC# ILotRepository.cs
└─ IC# IMediaObjectRepository.cs
└─ IC# IMessageRepository.cs
└─ IC# IModelRepository.cs
└─ IC# IModificationRepository.cs
└─ IC# INotificationRepository.cs
```

Інтерфейси

5

## Загальні правила форматування коду

Будь-які скоупи мають відділяти власний скоуп, починаючи з фігурної дужки на наступному рядку, одна табуляція для подій всередині, та з нового рядку фігурна дужка на рівні першої фігурної дужки

```
if (mediaObject == null)
{
    return false;
}
```

Правильно

```
if (mediaObject == null) {
    return false;
}
```

Не правильно

6

## Загальні правила форматування коду

Також варто зазначити, що існує необхідність виділяти рядки логічних операторів, циклів, або за логікою в середині самих рядків

7

```
[HttpGet("GetListLotViewModel")]
0 references
public async Task<ActionResult> GetCars()
{
    var lots = await _lotService.Retrieve();
    var resultlist = new List<LotViewModel>(); } 1

    foreach (var lot in lots)
    {
        if (lot.LotStatus == LotStatus.OnAuction)
        {
            var lotViewModel = await _lotConverterService.ConvertToViewModel(lot);
            resultlist.Add(lotViewModel);
        }
    }

    return Json(resultlist);
}
```

Ось гарний приклад відділення коду. Тут можемо побачити, що код умовно ділиться на "секції". Перша секція - отримання даних. Друга секція - цикл. Третя - логічний оператор.

8

Джерела

[learn.microsoft.com/en-us/dotnet/standard/design-guidelines/naming-guidelines](https://learn.microsoft.com/en-us/dotnet/standard/design-guidelines/naming-guidelines)

[google.github.io/styleguide/csharp-style.html](https://google.github.io/styleguide/csharp-style.html)